KANTIPUR ENGINEERING COLLEGE

(Affiliated to Tribhuvan University)

Dhapakhel, Lalitpur

[Subject Code: CT755]

A MAJOR PROJECT PRE-FINAL REPORT ON

NETWORK INTRUSION DETECTION SYSTEM USING FAST KNN

Submitted by:

Pranav Neupane [KAN077BCT057]

Samrat Chaudhary [KAN077BCT073]

Shyama Mainali [KAN077BCT081]

Utsarga Regmi [KAN077BCT096]

A MAJOR PROJECT SUBMITTED IN PARTIAL FULFILLMENT OF THE REQUIREMENT FOR THE DEGREE OF BACHELOR IN COMPUTER ENGINEERING

Submitted to:

Department of Computer and Electronics Engineering

February, 2025

NETWORK INTRUSION DETECTION SYSTEM USING FAST KNN

Submitted by:

Pranav Neupane [KAN077BCT057]

Samrat Chaudhary [KAN077BCT073]

Shyama Mainali [KAN077BCT081]

Utsarga Regmi [KAN077BCT096]

Supervised by:

Er.Suman Shrestha

System Admin

IT Department,Kantipur Engineering College

A MAJOR PROJECT SUBMITTED IN PARTIAL FULFILLMENT OF THE REQUIREMENT FOR THE DEGREE OF BACHELOR IN COMPUTER ENGINEERING

Submitted to:

Department of Computer and Electronics Engineering

Kantipur Engineering College

Dhapakhel, Lalitpur

February, 2025

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

LIST OF ABBREVIATIONS

ACK	Acknowledge
CICIDS	Canadian Institute for Cybersecurity Intrusion Detection System
CNN	Convolutional Neural Network
CSP	Constraint Satisfaction Problem
DoS	Denial of Service
DDoS	Distributed Denial of Service Attack
FKNN	Fuzzy K-Nearest Neighbors
IAT	Inter-Arrival Time
IDS	Intrusion Detection System
IPS	Intrusion Prevention System
KDD	Knowledge Discovery in Database
KNN	K-Nearest Neighbor
NDR	Network Detection Response
NIDS	Network Intrusion Detection System
NGF	Next-Generation Firewalls
NGIPS	Next-Generation Intrusion Prevention System
PDS KNN	Prototype Selection K-Nearest Neighbor
SVM	Support Vector Machine
TCP	Transmission Control Protocol
UDP	User Datagram Protocol
U2R	User To Root
VIPDS	Variable Improved Particle Swarm Optimization and Differential Evolution Strategy

CHAPTER 1
INTRODUCTION

1.1 Background

With the rapid expansion of computer usage and computer network the security of the computer system has become very important. Every day new kind of attacks are being faced by industries. As the threat becomes a serious matter year by year, intrusion detection technologies are indispensable for network and computer security. It is important to increase the detection rates and reduce false alarm rates in the area of network intrusion detection.Intrusion detection system (IDS) is an application that monitors network traffic and searches for known threats and suspicious or malicious activity. The IDS sends alerts when it detects any security risks and threats.

Intrusion Detection and Prevention Systems (IDS/IPS) have evolved significantly since their inception. In the 1970s and 1980s, early concepts and practical implementations like IDES emerged, focusing on statistical anomaly detection. The 1990s saw the commercialization of IDS, with the introduction of network-based and signature-based systems. From 2000 to 2005, IDS gained prominence due to emerging threats like SQL injections and XSS attacks, which bypassed firewalls. Organizations initially preferred IDS over IPS due to concerns about blocking legitimate traffic. Between 2006 and 2010, IPS throughput increased, and adoption grew, driven by PCI DSS requirements and enhanced detection capabilities. The 2011-2015 period saw the advent of next-generation IPS (NGIPS) with advanced features like application and user control, prompted by high-profile breaches like the RSA attack. From 2016 to 2020, major incidents such as WannaCry and SolarWinds highlighted the need for advanced solutions, leading to widespread adoption of next-generation firewalls (NGFW) with integrated IDS/IPS functionality.

Since 2021, the integration of machine learning, AI, and Network Detection and Response (NDR) solutions has further improved threat detection and response, adapting to modern challenges posed by IoT, remote work, and sophisticated cyber threats.The

algorithms such as KNN, decision tree, Random Forest and SVM are used in this field. Network Intrusion Detection System(NIDS) a type of IDS deployed at strategic points within an organization's network to monitor incoming and outgoing traffic. This IDS approach monitors and detects malicious and suspicious traffic coming to and going from all devices connected to the network [1].

1.2 Statement of Problem

Firewall and anti-virus won't be enough against any intrusion. Without a good detection system, a computer network will be access by an unauthorized individual. This individual may do harm to others by stealing other people's data not to mention confidential information would be compromise. A Denial of Service (DoS) attacks may also occur.

1.3 Objective

The main objective of this project can be listed as follows:

- To investigate the methods needed to detect any malicious attacks into a networking system.
- To analyze flow and features of captured packets.
- Utilizing MiniEdit to graphically display and manage network topology for better visualization and analysis.

1.4 Application and Scope

An Intrusion Detection System (IDS) is essential for modern cybersecurity, continuously monitoring network and system activities to detect malicious actions and policy violations in real time. It protects against various attacks, such as denial-of-service (DoS) and man-in-the-middle attacks, and ensures adherence to security policies. By analyzing behavioral patterns, IDS detects anomalies that might indicate security threats, including insider threats and compromised accounts. Integrating threat intelligence, IDS uses known signatures and advanced algorithms to identify both known and unknown threats. Overall, IDS is a versatile tool that enhances real-time monitoring,

incident response, compliance, and anomaly detection in various IT environments.

1.5 Features

This project possess various features depending upon its specific goals and requirement. It continuously monitors and analyzes network traffic for suspicious activities. Similarly, it identifies threats by matching traffic against a database of known attack signatures.It also supports anomaly-based detection which detects unknown threats by identifying deviations from normal network behavior. Then provides immediate alerts for quick response to detected threats. Integration with other security tools for comprehensive defense strategy.

1.6 Feasibility Study

A feasibility study analyses the viability of a project to determine whether the project or venture is likely to succeed. The feasibility analysis of the project is one of the most important things to be considered for the project development. It can be divided into the following components.

1.6.1 Technical Feasibility

Ensuring the technical feasibility of an Intrusion Detection System (IDS) involves several important factors. Firstly, the IDS must be capable of handling both current and future network traffic to ensure it can scale as the organization grows. It's crucial to evaluate the system's response time, detection accuracy, and resource usage to ensure it performs well under different conditions. Secondly, the IDS should be compatible with the existing network infrastructure, such as firewalls and routers, to integrate smoothly without causing issues. Additionally, the ease of deployment, configuration, and integration with current IT systems is essential, as these factors affect how quickly and effectively the IDS can be implemented.

1.6.2 Operational Feasibility

Evaluating the operational feasibility of an Intrusion Detection System (IDS) involves several key considerations. First, the ease of use of the IDS interface for security analysts is crucial, as a user-friendly interface can significantly enhance their efficiency and effectiveness. The system's capabilities for alert management and incident response must also be assessed to ensure that it can handle alerts promptly and provide comprehensive support during security incidents. Additionally, it's important to analyze the impact of IDS implementation on daily operations and existing workflows to identify any potential disruptions and ensure smooth integration. Ongoing maintenance needs, including regular updates and patch management, must be evaluated to ensure the IDS remains effective over time.

1.6.3 Economic Feasibility

Evaluating the economic feasibility of an Intrusion Detection System (IDS) involves two primary considerations. Firstly, estimating costs for acquiring and deploying the IDS includes factors such as purchasing hardware and software, licensing fees, and implementation expenses. Secondly, ongoing maintenance, update, and support costs must be considered, encompassing expenses for regular updates, patch management, vendor support, and staff training. By assessing both initial and ongoing costs, organizations can determine the economic viability of implementing an IDS.

1.6.4 Legal and Regulatory feasibility

Identifying potential legal risks and liabilities associated with IDS deployment is essential to mitigate them effectively. Assessing data handling and privacy measures helps protect sensitive information and ensure compliance. Implementation of robust safeguards, along with clear policies and procedures, is vital for regulatory compliance and data protection.

1.6.5 Schedule Feasibility

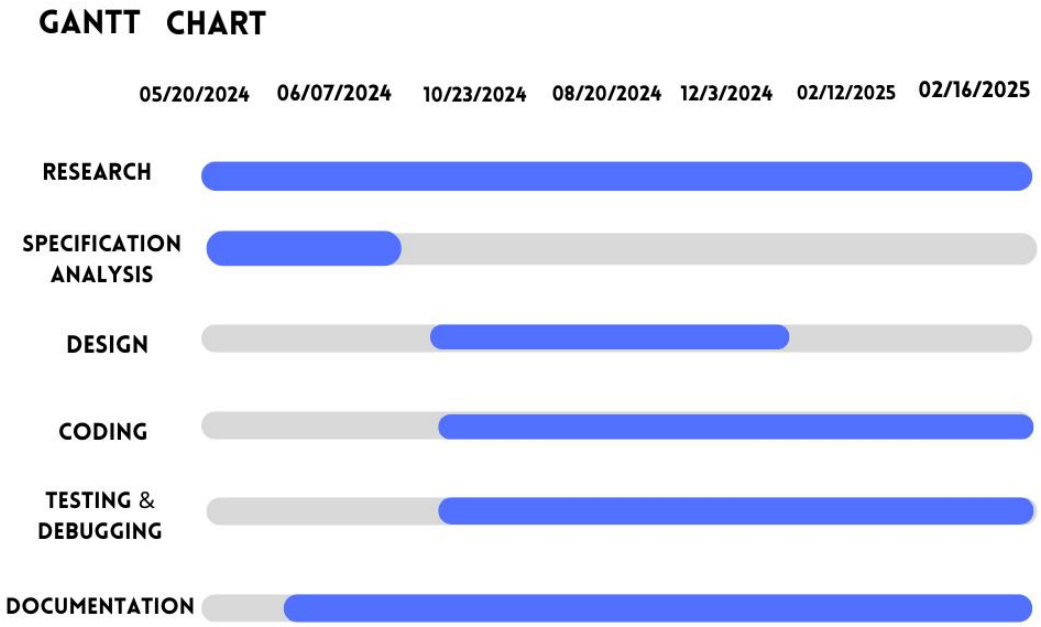

Figure 1.1: Gantt Chart

1.7 Project Requirements

1.7.1 Development Requirements

Table 1.1: Development Requirements

Hardware Requirements	Software Requirements
RAM:8GB and dedicated graphics card.	Operating system: Windows 10, 11 and Linux.
Intel i5 series processor or higher.	Web browser: Google Chrome, Opera.
	Environment and IDE: Google Collab

1.7.2 Deployment Requirements

Table 1.2: Deployment Requirements

Hardware Requirements	Software Requirements
PC with 8GB RAM.	Web browser: Google Chrome, Opera.
Intel i5 series processor.	Internet connection.

CHAPTER 2
LITERATURE REVIEW

2.1 Related Papers

This paper ”A Network Intrusion Detection Method Based on a fast KNN Algorithm” proposes a network intrusion detection method based on fast KNN algorithm. The algorithm proposed in this paper is improved in terms of efficiency and accuracy, and is validated on the KDD 99 dataset. As the original KDD CUP99 dataset is too large, only 80056 of them are randomly selected for the study, 40000 are used as training samples and 40056 were used as test samples. The structure of the intrusion detection model based on the fast KNN algorithm includes data preprocessing, feature reduction and classification. The paper introduces an optimized version of the KNN algorithm, termed Fast KNN, designed to address the computational inefficiencies of the traditional KNN algorithm. By incorporating continuous data discretization, character data digitization, and a feature reduction algorithm based on mutual information, the Fast KNN algorithm significantly improves classification efficiency and accuracy. The evaluation criteria include classification accuracy, misdetection rate and missed detection rate. The effectiveness of the Fast KNN algorithm is validated through experiments on the KDD CUP99 dataset, a benchmark for evaluating the performance of intrusion detection systems. The results, as presented in the paper, show a marked improvement in classification accuracy and speed compared to both the traditional KNN algorithm and other existing technologies. Specifically, the Fast KNN algorithm demonstrates higher classification accuracy with significantly reduced false detection rates and computational time[2].

This paper ”A Novel Framework for NIDS through Fast KNN Classifier on CICIDS2017 Datase”investigates the performance of a Fast k-Nearest Neighbor Classifier(FkNN) for Network Intrusion Detection System(NIDS) on cloud environment.A benchmark dataset CICIDS2017 is considered for the evaluation process because it is a 78 featured dataset with most updated cloud related attacks.CICIDS2017 dataset contains benign and the most up-to-date common attacks, which resembles the true real-world data.The

work in this paper is based on the KNN classifier and its variations.In the first phase only normalization is carried out because all the features in the dataset are numeric and there is no need of transformation.In this paper for normalization process the min-max normalization technique is used. The methodology of the proposed framework includes data preprocessing, variance based feature indexing and classification. After the implementation of KNN, PDS-KNN and FkNN for different values of the k(3,5 and 7) with 10 fold cross validation,the result regarding the Accuracy, precision, Recall and computational time was obtained. The computational time is measured based on the execution time of the classifier. The FkNN is compared with kNN and PDS-kNN in terms of Accuracy, Precision, Recall and computational time.It is concluded that the FkNN classifier is a better classifier with less detection time and without loss in Accuracy. The gain in computational time is more than 88% as compared to traditional kNN. The authors suggested FkNN as a better machine learning algorithm for NIDS with latest attack types and with high dimensional dataset for cloud environment to mitigate attacks and reduce the economic losses of CSPs with less span of time [3].

This paper"Intrusion Detection System classification using different Machine learning Algorithms on KDD-99 and NSL-KDD datasets"experiments the performance of the different machine learning algorithms using KDD-99 Cup and NSL-KDD datasets.KDD-99 dataset is a multi-variety dataset. The characteristics of the attributes used in the dataset are categorical and integer in nature. To solve the issues in KDD-99 cup dataset, reseachers proposed NSl-KDD datasets which consists of only selected records from the complete KDD dataset. This KDD dataset consists of mainly four types of attack which includes Denial of Service(DOS),Remote to Local(R2L),User to Root(U2R) and Probing. This paper have introduced an overview of different machine learning algorithms for Intrusion Detection System (IDS) which includes Supervised learning, Semi-Supervised learning and Un-supervised learning. In case of Supervised Learning, +1 is used for normal data and -1 is used for attacked data. The evaluation of various algortihm reveals distinct trade-offs between accuracy and false alarm rates. Random forest stands out with the highest accuracy at 99.0% and a low false alarm rate at 2.43 making it highly suitable for applications demanding both metrics in the context of KDD-99 cup dataset.Also in case of NSL-KDD dataset, the random forest algorithm achieves

the highest accuracy at 99.7% with false alarm rate of 3.2[4].

This paper ”Network Intrusion Detection System”presents a timely exploration into the realm of cybersecurity, specifically focusing on the development and implementation of an Intrusion Detection System (IDS) that employs machine learning algorithms to safeguard data transmitted over the Internet. This study is set against the backdrop of an ever-increasing volume of data being transmitted over the Internet, which has escalated the need for robust security measures to detect and mitigate potential cyber threats effectively. The authors of this paper have embarked on a project to enhance the capabilities of IDS by incorporating advanced machine learning techniques, namely Support Vector Machine (SVM) and Naive Bayes algorithms. These algorithms are chosen for their proficiency in handling classification problems, which are central to the operation of an IDS. By employing the NSL-KDD knowledge discovery dataset, the paper evaluates the performance of these algorithms in detecting various types of network intrusions, including Denial of Service (DoS) attacks, User to Root (U2R) attacks, Remote to Local (R2L) attacks, and Probe attacks. One of the paper's key findings is the superior performance of the SVM algorithm over Naive Bayes in terms of accuracy and misclassification rates. This insight is particularly valuable as it underscores the importance of selecting the most effective machine learning model for the task of intrusion detection. Furthermore, the study explores the potential integration of an Intrusion Prevention System (IPS) to further bolster network security, suggesting the use of additional machine learning models like Convolution Neural Network (CNN) and Support Vector Network (SVN) for future research. The paper provides a comprehensive overview of existing IDS technologies, including the use of honeypots and Snort, an open-source network intrusion detection tool. It also categorizes IDS into five distinct types, each with specific functionalities and applications, offering readers a broad understanding of the current landscape in network security technologies[5].

The paper ”Network Intrusion Detection Based on LSTM and Feature Embedding” proposes a novel approach to intrusion detection in networks, leveraging Long Short-Term Memory (LSTM) networks. The system detects a variety of attacks, including DoS (Denial of Service), Probe, U2R (User to Root), and R2L (Remote to Local) attacks. The researchers used the NSL-KDD dataset for experimentation. The study uses fea-

ture embedding to enhance feature representation, making it easier for the model to understand and detect complex patterns in network data. The system's output demonstrated high accuracy in detecting various attacks, outperforming traditional intrusion detection methods. Using LSTM allowed the model to better understand the sequential nature of network traffic, leading to improved detection rates. This methodology also showed that the combination of feature embedding and LSTM significantly enhanced the model's performance in identifying complex attack patterns. In conclusion, the research shows that using LSTM-based models, along with effective data preprocessing and feature embedding, can provide a robust solution for network intrusion detection, capable of identifying various types of malicious activities in a network with high accuracy of 83% in multi-classification[6].

The paper titled "Network Intrusion Detection based on LSTM" explores the application of Long Short-Term Memory (LSTM) networks in enhancing the accuracy and response speed of network intrusion detection systems (NIDS). As cyber threats continue to evolve, traditional intrusion detection methods face challenges such as gradient vanishing and exploding issues inherent in standard recurrent neural networks (RNNs). The authors utilize the KDD99 dataset, a well-established benchmark for intrusion detection, to train their LSTM model. Through a comprehensive methodology that includes data preprocessing, model design, and evaluation, the study demonstrates that the LSTM model achieves an impressive accuracy of 88.01% in detecting network intrusions. The research emphasizes the importance of deep learning techniques in autonomously identifying malicious behavior by analyzing network traffic data over time. The paper also discusses the architecture of the LSTM model, highlighting its components such as input gates, forget gates, and output gates, which collectively enhance its ability to learn from sequential data. Furthermore, the authors address the significance of loss functions and optimization techniques like gradient descent in improving model performance. The findings suggest that LSTMs can effectively improve NIDS capabilities, making them a valuable tool in the ongoing battle against cyber threats. Future work is proposed to explore more diverse datasets and hybrid models to further enhance detection accuracy and robustness[7].

2.2 Proposed System

The proposed system aims to significantly enhance network security by employing machine learning algorithms to monitor network traffic for suspicious activities and issue real-time alerts upon detection of potential threats. The system architecture includes key components such as data collection, preprocessing for data cleaning and normalization, and feature extraction to highlight critical data characteristics. Machine learning models, specifically K-Nearest Neighbors (KNN) and Fuzzy K-Nearest Neighbors (FKNN), will be trained using the CICIDS 2017 dataset to detect anomalies and known threats through anomaly-based detection methods. Upon identifying suspicious activities, the system will generate detailed alerts and implement automated responses to mitigate threats. Reporting will support network administrators in analyzing detected intrusions and maintaining security protocols. The implementation plan involves phases of setup, data collection, preprocessing, model training, system integration ensuring thorough testing and optimization at each stage. This system will empower network analysts to effectively monitor and secure their networks, providing valuable insights into traffic behaviors and emerging cyber threats, thereby demonstrating the crucial role of machine learning in advancing network security.

CHAPTER 3
METHODOLOGY

3.1 Working Mechanism

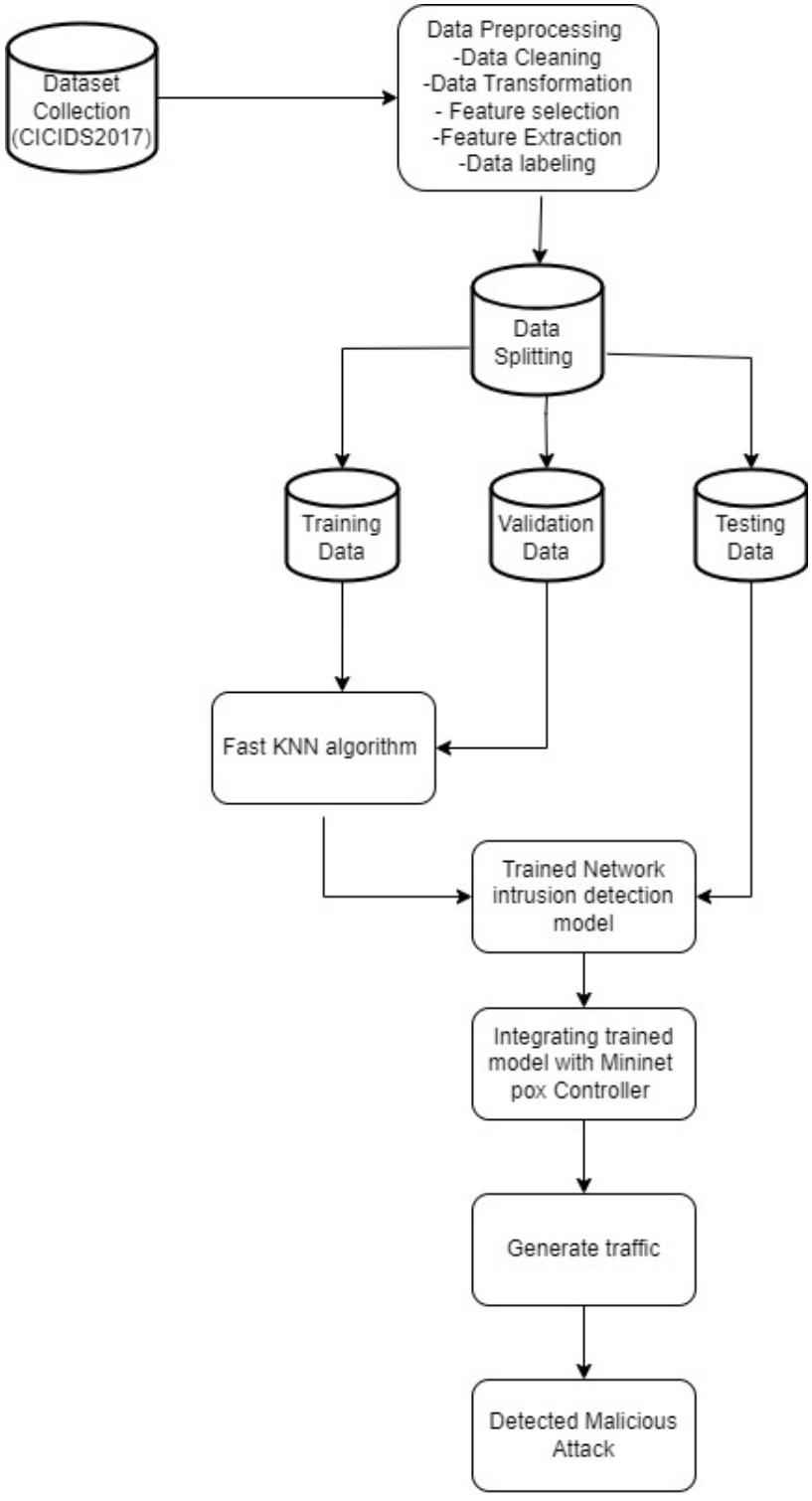

Figure 3.1: Block diagram of NIDS system

The Network Intrusion Detection System(NIDS) uses the machine learning approach to detect the threats and suspicious or malicious activities. The process will start by choosing the dataset. Further, pre processing is done. Then the data set is split to training,validating and testing set.The model is trained using the KNN algorithm .The trained model undergoes evaluation to assess its performance on validation data. Then in the testing phase the attacks are sent to the system and checked if it can detect it correctly in the correct time or not.

3.1.1 Dataset Description

The dataset is collected based on the types of attacks to be detected. As the project aims to detect the DDos and heart bleed attacks in the particular network.'CICIDS2017' is one of the most appropriate dataset for those attacks. CICIDS2017 dataset contains benign and the most up-to-date common attacks, which resembles the true real-world data (PCAPs). It also includes the results of the network traffic analysis using CICFlowMeter with labeled flows based on the time stamp, source, and destination IPs, source and destination ports, protocols and attack (CSV files). Also the extracted features are available. The features are:

.

No.	Feature Name
1	bwd_payload_bytes_mean
2	payload_bytes_variance
3	bwd_packets_length_total
4	fwd_packets_length_total
5	fwd_packets_IAT_std
6	fwd_payload__
7	fwd_payload_bytes_max
8	bwd_total_header_bytes
9	fwd_total_header_bytes bwd_packets_rate
10	packets_rate bwd_init_win_bytes
11	fwd_packets_IAT_mean fwd_init_win_bytes
12	Label

Table 3.1: Features of CICIDS2017 datset

3.1.2 Network Infrastructure

Network Infrastructure refers to any source that can feed the system with raw TCP and UDP packets. It can be a live computer network, any specific device generating network traffic or an offline source of packet data. This subsystem can be said to be the infrastructure under analysis for intrusion. For best results, it is desired to be the gateway, so that maximum traffic can be interpreted.

3.1.3 Data preprocessing

After the selection of the appropriate dataset we move to pre-processing which includes: Data cleaning, feature extraction, normalization, handling the categorical and imbalanced data and Traffic Segmentation. The Data preprocessing is essential to ensure that the data is in a suitable format for training and testing intrusion detection models. It provides the output with least amount of errors.

3.1.4 Intrusion Detection Module

This is the core of the system, responsible for classifying the flows captured as normal or anomalous. During training phase of the system, a Random forest classifier model and a K nearest Neighbors classifier model are trained and validated with the CICIDS2017 dataset. Those models are used to predict the behavior of any traffic flow captured.

KNN classification

Fast KNN uses the CICIDS2017 dataset to detect network attacks through a well-defined process. First, each network packet or flow is converted into a feature vector, which includes metrics like flow duration, the number of forward and backward packets, packet length statistics (mean, maximum, minimum), and details about flags and protocols. These feature vectors are then used as inputs for the Fast KNN algorithm.

In the training phase, a subset of the CICIDS2017 dataset is used, containing both

normal and attack traffic labeled with their respective types. Fast KNN calculates the distances between these feature vectors using metrics such as Euclidean distance. To speed up this process, Fast KNN employs optimizations like efficient indexing and approximate nearest neighbor techniques.

When a new packet or flow arrives, it is converted into a feature vector, and Fast KNN identifies the 'K' nearest neighbors by comparing this new vector to those in the training set. Each of these K neighbors votes on the classification of the new packet. The majority vote determines whether the packet is classified as normal or as one of the attack types, such as DDoS, DoS, or Probe.

$$d(\mathbf{v}_{\text{new}}, \mathbf{v}_i) = \sqrt{\sum_{j=1}^{m} (v_{\text{new},j} - v_{i,j})^2} \tag{1}$$

where:

- $\mathbf{v}_{\text{new}}$ is the feature vector of the new network flow.
- $\mathbf{v}_i$ is the feature vector of the i-th flow in the training set.
- $v_{\text{new},j}$ and $v_{i,j}$ are the j-th feature values of $\mathbf{v}_{\text{new}}$ and $\mathbf{v}_i$, respectively.
- m is the number of features used in the dataset (e.g., flow duration, total packets, packet length statistics).

3.1.5 Attacks

DOS attacks

In the DoS attack, Netcat is used to repeatedly transmit messages from host 2 to host 1 over port 8123. The loop generates a large number of messages, which are piped into Netcat, allowing continuous data transmission. Since Netcat operates as a simple but efficient networking utility, it sends the messages without requiring a formal connection, making it useful for testing and attack simulations. This overwhelms host 1 by consuming network bandwidth and processing power, potentially slowing down or

crashing any service running on port 8123 due to excessive incoming traffic.

DDOS

DDoS attack is sent by making multiple attacker hosts send traffic to a single victim host 3, over port 8123 using Netcat . Each attacker runs a loop to generate messages. These messages are piped into Netcat, which transmits them to host 3. Since multiple attackers execute this command simultaneously, host 3 is flooded with high-volume traffic from multiple sources, making it harder to filter or block. This can consume network bandwidth, overload system resources, and potentially crash services running on port 8123, disrupting normal operations.

3.1.6 Simulation of attacks using Mininet

Mininet is typically installed on a Linux-based system, often using Ubuntu. The installation process includes all the necessary dependencies and tools. Once installed, Mininet can be started by running predefined commands or custom Python scripts, which initializes the Mininet environment for creating and managing virtual networks.

After setting up Mininet, the POX controller is configured to run in the background. A custom topology is then created, defining the connections between hosts (h1, h2, h3, h4, h5, h6, h7) and switches(s1 and s2). In this setup, Host h2 initiates a Denial of Service (DoS) attack against Host h1, while Hosts h4, h5, h6, and h7 launch Distributed Denial of Service (DDoS) attacks against Host h3.This packets are captured using tcpdump command and stored in pcap file format.

The network traffic generated during these attacks is captured using Wireshark. Features from this captured traffic are then extracted using NTLflowlyzer, and these features are sent for prediction using a fast KNN model. Finally, MiniEdit is utilized to overview the network topology graphically.

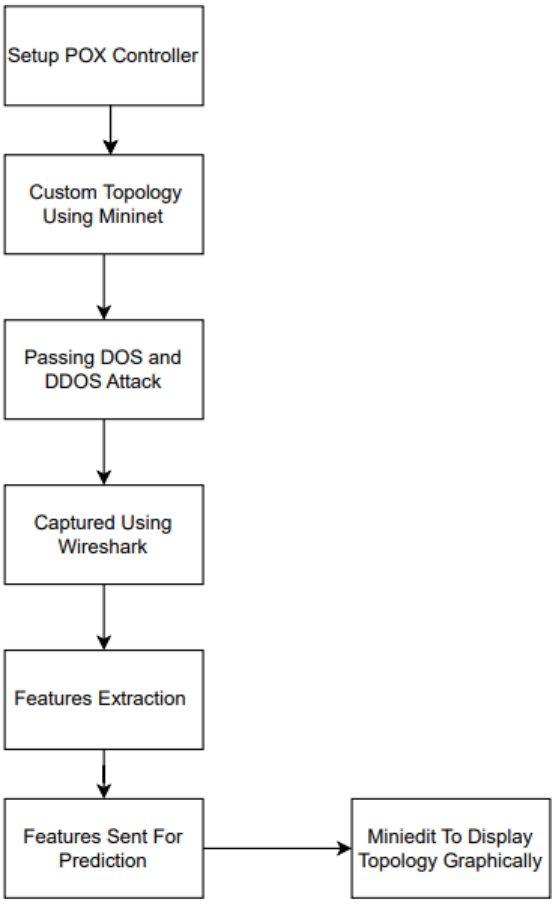

Figure 3.2: Block diagram of Simulation of attacks using Mininet

3.1.7 Feature Extraction using NTLFlowlyzer

NTLFlowLyzer generates bidirectional flows from the Network and Transportation Layers of network traffic, where the initial packet establishes both the forward (source to destination) and backward (destination to source) directions. This enables the calculation of statistical time-related features separately for each direction. Additional features include the ability to select from existing features, introduce new ones, and control the flow timeout duration [**?**]. From the generated pacap file from tcpdump, features are extracted and stored in csv format using NTLFlowlyzer. The extracted features are:

3.2 Devlopment Model

3.2.1 Incremental Model

This network intrusion detection system follows an incremental software development model, where each stage builds upon the previous one, adding new features and improvements iteratively.

In the first increment, we selected the dataset and focused on simulating malicious attacks using Mininet while capturing and analyzing them through Wireshark, laying the foundation for data collection and analysis. In the second increment, we preprocessed the selected dataset to ensure it was properly filtered and normalized for effective analysis. We then trained the Fast KNN (FKNN) model to classify and detect various network intrusions.

In the third iteration, we designed a customized network topology and executed DoS and DDoS attacks between hosts. We captured the attack packets using Wireshark and extracted relevant features using NTLflowlizer. The fourth iteration involved enhancing the FKNN model for more accurate predictions and feeding the extracted features into it. Additionally, we visualized the network topology in MiniEdit for better clarity and analysis.

This incremental approach ensures that each component is thoroughly tested and optimized before progressing, ultimately resulting in a robust and resilient network intrusion detection system.

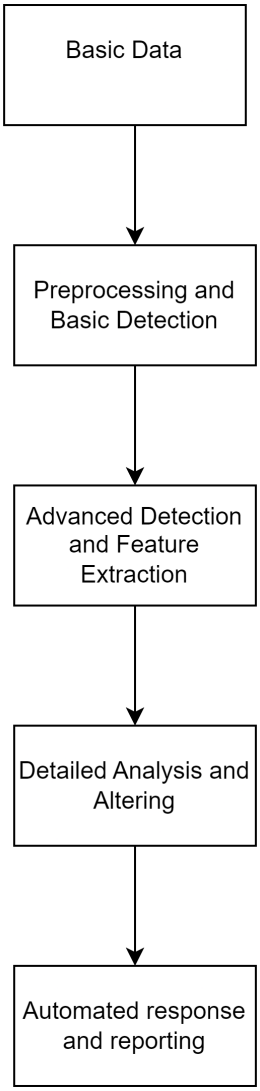

Figure 3.3: Incremental Model

3.3 Use Case Diagram

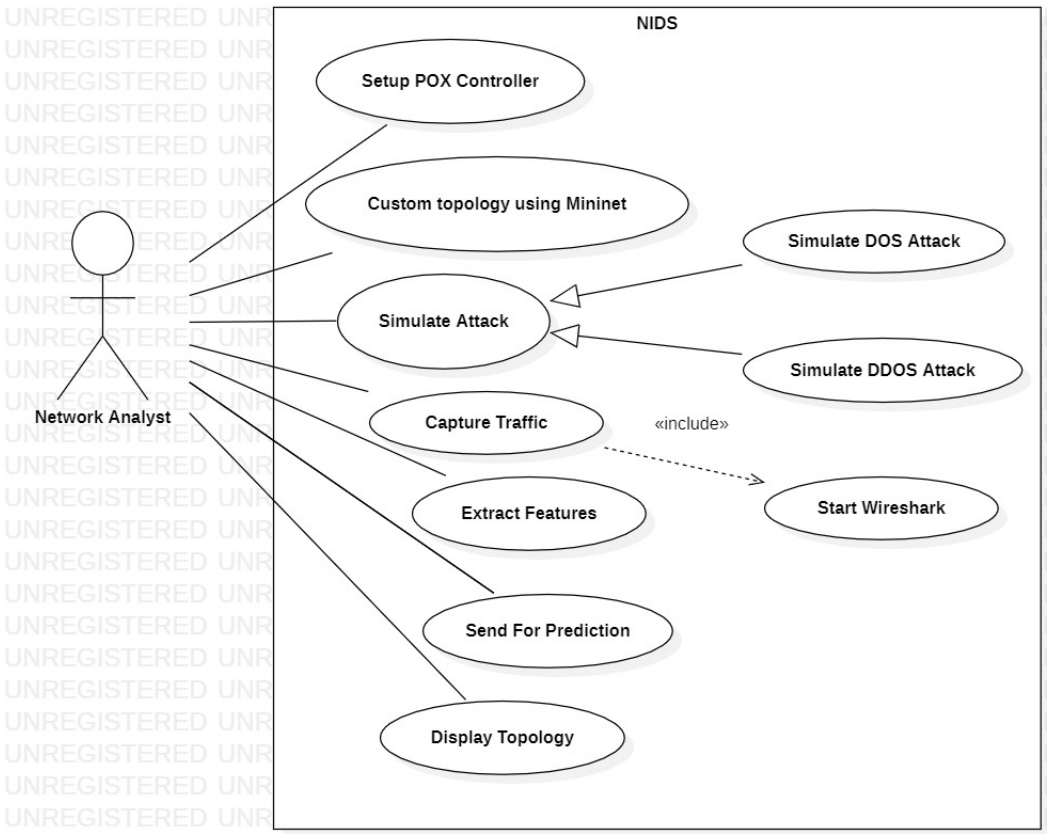

Figure 3.4: Use Case Diagram of NIDS

3.4 Activity Diagram

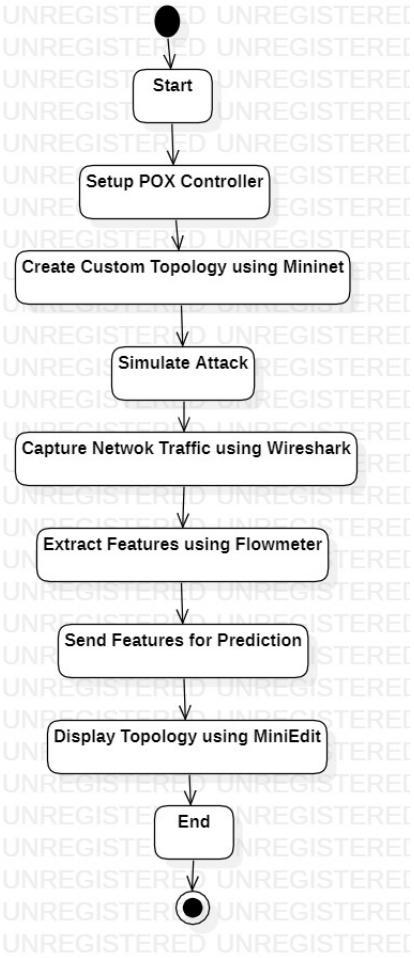

Figure 3.5: Activity Diagram of NIDS

CHAPTER 4
EPILOGUE

4.1 Work Done

4.1.1 Preprocessing

After selecting the CICIDS2017 dataset, preprocessing steps were carried out. First, a check for any missing or null values was performed, and it was confirmed that no null values were present in any of the labels. Next, the correlation between features was analyzed using a correlation matrix, and highly correlated features were removed to reduce overfitting, improve model efficiency, and assist with dimensionality reduction. Under-sampling was then applied to reduce the dataset size from 2.7 million to 1.5 million samples, making it more manageable for training. To address the issue of class imbalance, SMOTE (Synthetic Minority Over-sampling Technique) was used to balance the distribution of classes. Finally, the data was normalized to ensure that all features were on a similar scale, preventing any feature from being unfairly weighted due to its larger scale and ensuring fairness in the model's training process.

4.1.2 Visualization

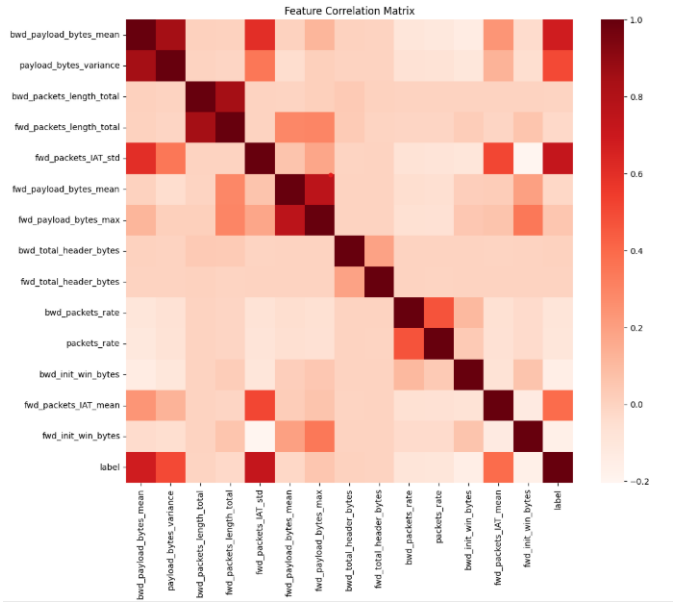

Figure 4.1: Correlation matrix of highly correlated features

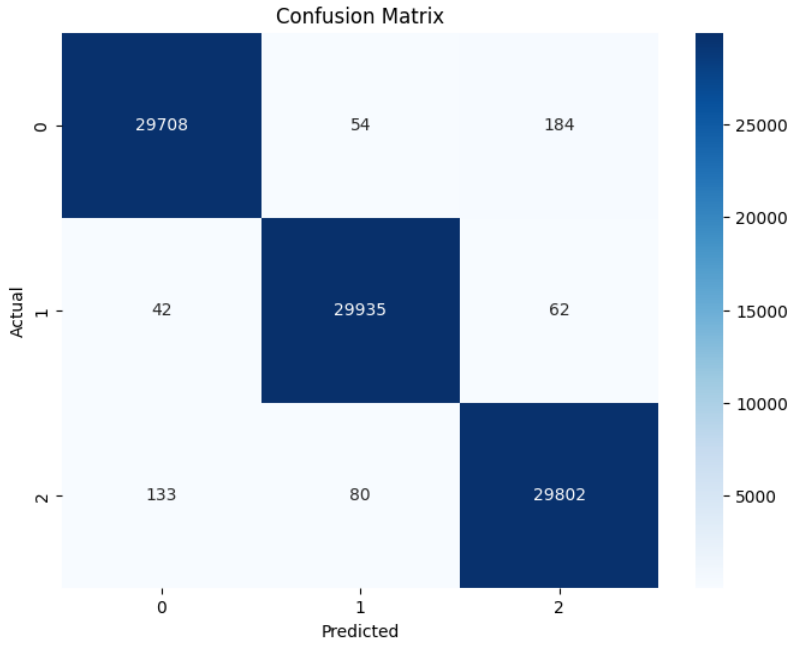

Figure 4.2: Confusion matrix

4.1.3 Model Training

Fast-KNN

- Data Preparation: Features (X) are all columns except the last; Target (y) is the last column containing outcomes.
- Dataset Splitting: Split data into training (80%) and testing (20%) sets using train_test_split.
- Model Initialization: Initialize KNeighborsClassifier with n=20,algorithm:'ball tree', and default settings for weights and distance me+ic.
- Training: Fit the model using the training data.
- Prediction: Predict test labels by finding nearest neighbors.

4.1.4 Output

	precision	recall	f1-score	support
Benign	0.99	0.99	0.99	29946
DDoS	1.00	1.00	1.00	30039
DoS	0.99	0.99	0.99	30015
accuracy			0.99	90000
macro avg	0.99	0.99	0.99	90000
weighted avg	0.99	0.99	0.99	90000

Figure 4.3: Performance Metrices

4.1.5 Live attack passing through Mininet

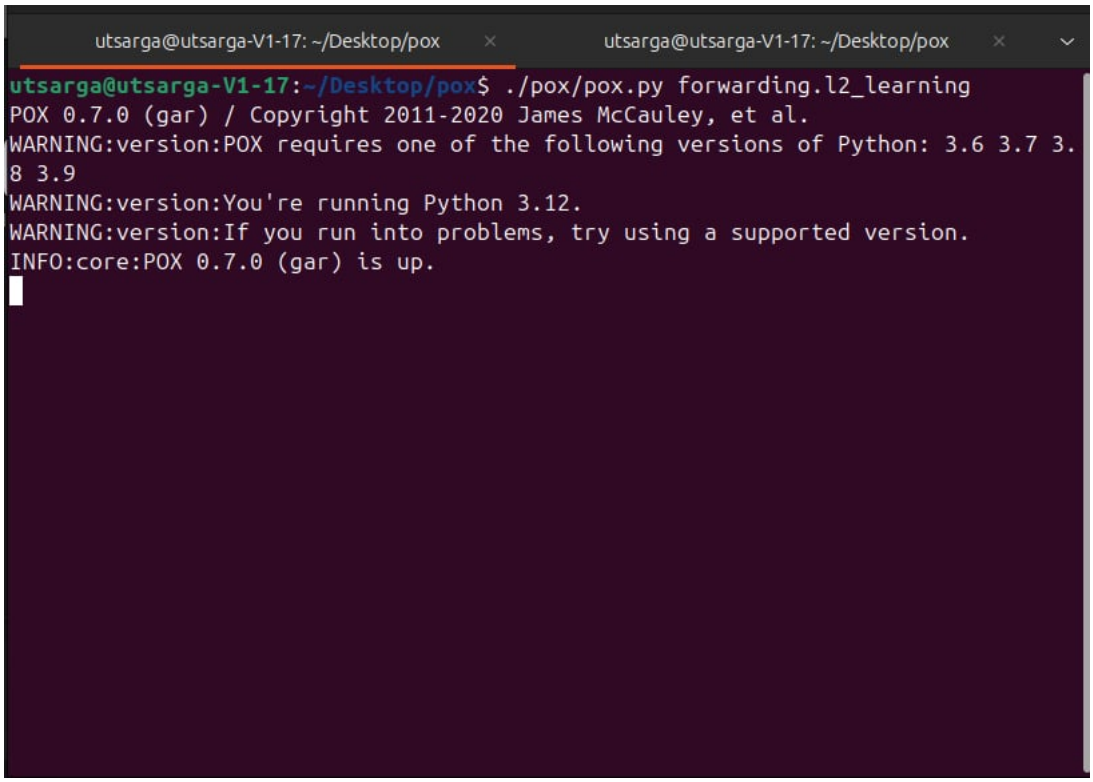

Figure 4.4: Running the POX controller with a simple Layer 2 learning switch component.

```
*** Creating network
*** Adding controller
*** Adding hosts:
h1 h2 h3 h4 h5 h6 h7
*** Adding switches:
s1 s2
*** Adding links:
(s1, h1) (s1, h2) (s1, s2) (s2, h3) (s2, h4) (s2, h5) (s2, h6) (s2, h7)
*** Configuring hosts
h1 h2 h3 h4 h5 h6 h7
*** Starting controller
c0
*** Starting 2 switches
s1 s2 ...

*** Network is running. Testing connectivity...
*** Ping: testing ping reachability
h1 -> h2 h3 h4 h5 h6 h7 nat0
h2 -> h1 h3 h4 h5 h6 h7 nat0
h3 -> h1 h2 h4 h5 h6 h7 nat0
h4 -> h1 h2 h3 h5 h6 h7 nat0
h5 -> h1 h2 h3 h4 h6 h7 nat0
h6 -> h1 h2 h3 h4 h5 h7 nat0
h7 -> h1 h2 h3 h4 h5 h6 nat0
nat0 -> h1 h2 h3 h4 h5 h6 h7
*** Results: 0% dropped (56/56 received)
```

Figure 4.5: .
Creation of network topology

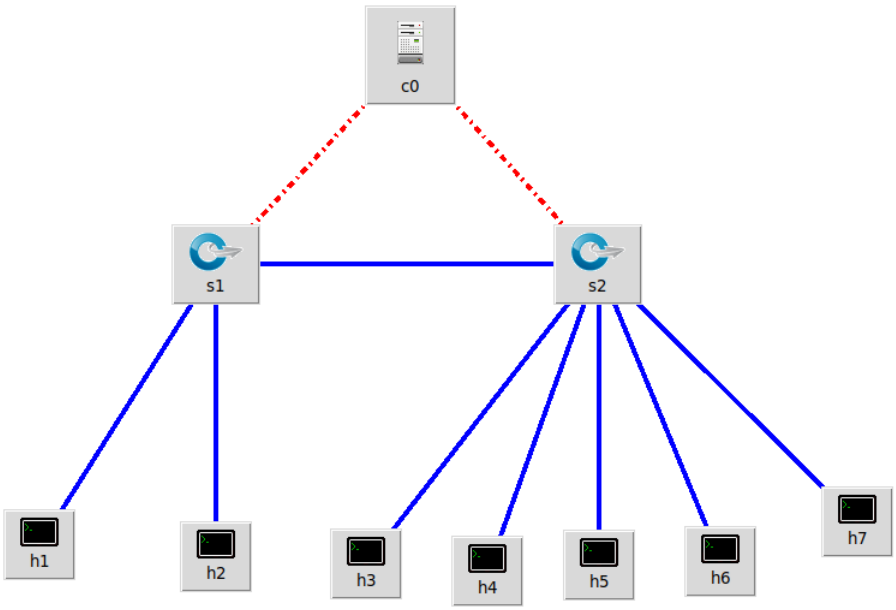

Figure 4.6: Network Topology using MiniEdit

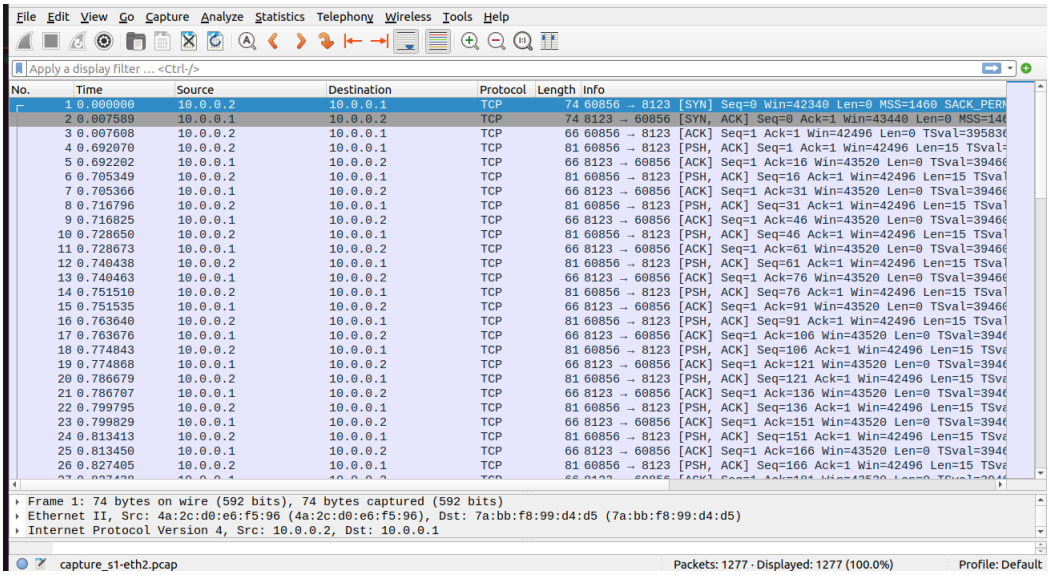

Figure 4.7: DOS attack form host h2 to host h1

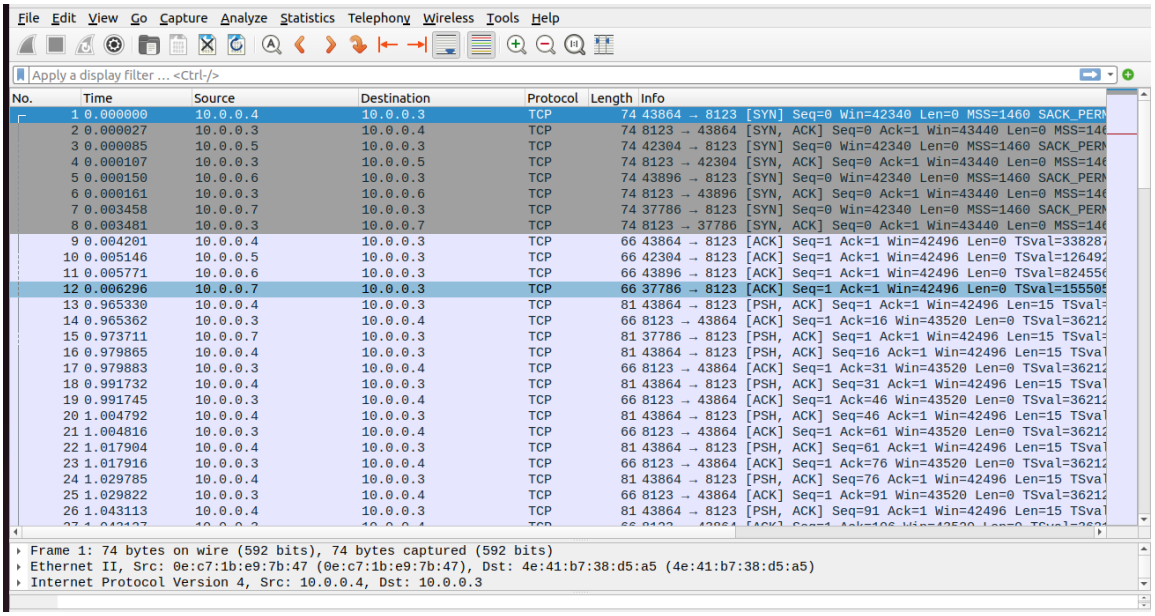

Figure 4.8: DDOS attack from host h4,h5,h6,h7 to host h3

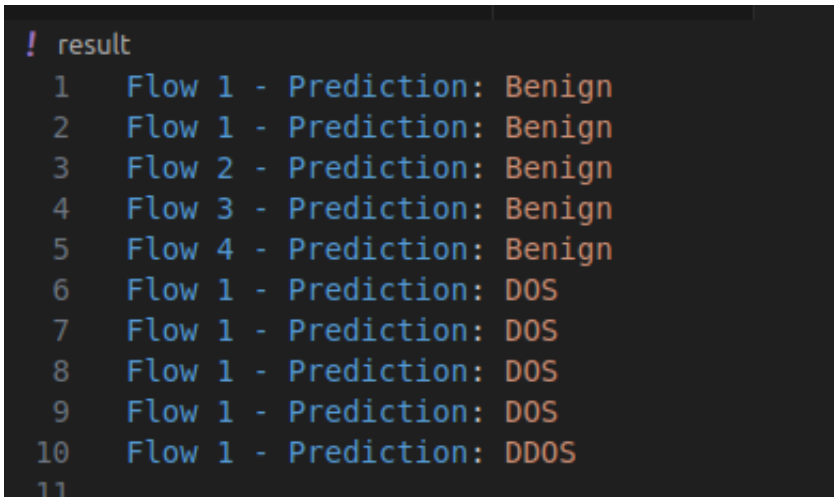

Figure 4.9: Live data prediction

CHAPTER 5
CONCLUSION AND FUTURE ENHANCEMENTS

This project successfully developed a Network Intrusion Detection System (NIDS) using the CICIDS2017 dataset and a Fast k-NN model to classify network traffic as DoS, DDoS, or benign. We improved detection accuracy by selecting the most significant features through feature importance scoring and reducing dimensionality with PCA. To create a realistic attack environment, we used Mininet with a POX controller and OVS switch to simulate DoS and DDoS attacks between hosts. The network was visualized using MiniEdit, and traffic flows were captured using NTLFlowlyzer, allowing us to extract key features for classification. This approach effectively demonstrates the use of machine learning and software-defined networking (SDN) for detecting cyber threats.

For further enhancements, we can expand detection capabilities beyond TCP by including UDP and ICMP traffic, making the system more robust against a wider range of attacks. Additionally, incorporating more attack types and increasing the scale and complexity of the virtual network will improve adaptability to real-world scenarios. Advanced techniques, such as deep learning models and adaptive anomaly detection, can also be integrated to enhance detection efficiency. These improvements will strengthen the system's ability to detect and respond to evolving cyber threats in real time.

REFERENCES

[1] "What is Intrusion Detection Systems (IDS)? How does it Work? — Fortinet — fortinet.com," https://www.fortinet.com/resources/cyberglossary/intrusion-detection-system?fbclid=IwAR362GmmY2RVCO8BPgQyCiMrz9Yg_hW6PriVEEa266qa6wzQffnD4XQOU7E#:~:text=An%20intrusion%20detection%20system%20(IDS)%20is%20an%20application%20that%20monitors,any%20security%20risks%20and%20threats, [Accessed 05-06-2024].

[2] P. He, G. Feng, H. Li, and L. Yang, "A network intrusion detection method based on a fast knn algorithm," *Academic Journal of Science and Technology*, vol. 8, no. 2, pp. 81–83, 2023.

[3] K. V. Krishna, K. Swathi, and B. B. Rao, "A novel framework for nids through fast knn classifier on cicids 2017 dataset," *International Journal of Recent Technology and Engineering (IJRTE)*, vol. 8, no. 5, pp. 3669–3675, 2020.

[4] R. D. Ravipati and M. Abualkibash, "Intrusion detection system classification using different machine learning algorithms on kdd-99 and nsl-kdd datasets-a review paper," *International Journal of Computer Science & Information Technology (IJCSIT) Vol*, vol. 11, 2019.

[5] P. Desai, A. Sonawane, T. Mane, and R. Jaiswal, "Network based intrusion detection system."

[6] H. Gwon, C. Lee, R. Keum, and H. Choi, "Network intrusion detection based on lstm and feature embedding," *arXiv preprint arXiv:1911.11552*, 2019.

[7] Y. Wu, B. Zou, and Y. Cao, "Current status and challenges and future trends of deep learning-based intrusion detection models," *Journal of Imaging*, vol. 10, no. 10, p. 254, 2024.